R E M A R K S

Applicant has carefully considered the Office Action of February 23, 2007, rejecting all of the claims. The present response is intended to fully address all points of objection raised by the Examiner, and is believed to place the application in condition for allowance. Favorable reconsideration and allowance of the application are respectfully requested.

Claims **1, 2, 15, 17, 21, 24** and **27** have been amended. Claim **30** has been added. Therefore, claims **1-30** remain in the case.

The present invention comprises a combination of three major elements:

1) visualizable computer executable modeling language for the definition of software solutions;
2) a modeling environment for visually defining the software solutions in the modeling language; and
3) a runtime engine that executes solutions defined in the modeling language.

As stated at the end of paragraph 0082 of the published application, the method of the present invention effectively achieves elimination of the writing of source code for developing software applications.

The Examiner has objected to the drawings under 37 C.F.R. 1.83(a) as failing to show every feature of the invention specified in the claims. The highlighted portions in claims **1, 2** and **27** are objected to for **not** being seen in the drawings:

Claim **1**:

"defining each of said flow rules as connecting a pair of said slots, data models and sub data models, wherein said flow rules define both data flow and process flow."

Accordingly, Figs. 4-5 and 7-14 have been amended to add reference numbers to the drawings, such as in Fig. 4, the reference number 411 has been appended to the arrow connecting point "In 410" to point "In 2 422," thereby CONNECTING two objects. Thus, in claim **1**, the words "CONNECTING a pair of said slots," is now reflected as a component of the drawing.

Flow rules define both data flow and process flow. Reference number 1127 has been added to Fig. 11, as an example defining the process flow, i.e., invocation of "Trade" rather than "Handle Exception." At the same time 1127 defines the data flow, wherein multiple instances of "Confirmation" 1118 are transferred through 1127 to the input slot 1124 of "Trade." Fig. 13 has similar examples.

Claim **2.**

"wherein said process models and data models and slots and flow rules are arranged in a structural hierarchy conforming to a set of rigid composition rules, ensuring the language system ~~is rich enough and precise enough for~~ enables a computer to execute an application model defined in said modeling language."

The drawings show that the structural hierarchy conforms to a set of rigid composition rules. This is shown as follows: Fig. 1a is an application object diagram showing several "sub data model(s) within (a) data model." Figs. 1b and 2 show several "slot(s) within (a) process."

Fig. 6 shows multiple flows to a repetitive trigger. Fig. 7 shows a flow 726 from a repetitive sub data model 725 to a non-repetitive trigger 731 of a repetitive process 730. Fig. 9 shows a flow 925 from an input slot of a parent process ("In" of "Handle Message") to an input slot of a child process ("In 2" of "Parse Message"). Fig. 12 shows a flow 1235 from a repetitive exit ("Out" of 1230) to a non-repetitive trigger of a repetitive process ("In 1" of 1240).

Claim **27.**

"defining each of said composite process models as a construction of at least one of sub process models, slots, data models and flow rules"

Fig. 3 illustrates the definition of composite process models as a construction of sub process models, data models and

flow rules. Fig. 3 shows a process model ("Handle_Header") containing a sub process model ("Concat"). Fig. 3 also shows a process model ("Handle_Header") containing a data model ("Msg 1").

Fig. 4, among others, also shows sub process models and data models within a process model. Many Figs. show flow rules, as described above with reference to claims 1 and 2.

Additionally, Fig. 12 was corrected to add the number **1235** which was described in the text at paragraph 145.

The Examiner has rejected claims **2-29** under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter Applicant regards as the invention. Accordingly, the relative term "substantially" has been deleted from claims **2, 17, 21, 24** and **27** and the relative term "virtually" has been deleted from claims 15. Also the phrase "rich enough and precise enough" has been deleted from claim **2**. Therefore, increased clarity and definition have been achieved to overcome this rejection.

The Examiner has rejected claims **1-13** and **21-23** under 35 U.S.C. 101 as being directed to non-statutory subject matter. The Examiner has also stated that the lack of hardware renders these claims intangible.

Claim 1 has been amended to add steps relating to the provision of hardware elements as follows: "providing a plurality of display elements for displaying screen objects on a display screen; and displaying components corresponding to said screen objects."

The addition of hardware elements to claim **1** overcomes the rejection since the method integrates the interaction of hardware and software elements, producing a tangible result.

It is the Applicant's position that the current amendments to claim **1** provide understanding of all claims **1-29** and provide inventive claim formulations of statutory subject matter, so the

basis of the Sec. 101 rejection is improper, and Applicant respectfully requests that it be withdrawn.

New claim **30** has been added drawn to a computer usable medium encoded with a computer program to achieve the goal of the inventive method, producing a tangible result.

The Examiner has rejected claims **1-12** and **14-16** under 35 U.S.C. 102(b) as being anticipated by Williams (US Patent No. 5,850,548), relative to column 2, lines 20-39, disclosing a visual programming environment based on a high level hierarchical data flow model.

Williams discloses that components communicate by sending and receiving "messages" on their "ports." Components can be constructed with a conventional programming language, such as C++ or Pascal, or constructed with Monet--the visual programming language of the system.

Williams further discloses that various components are linked together by establishing "connections" between their ports. When a component sends a message out on a connected port, the message is received by the component(s) at the other end of the connection. In response to the message, the receiving component is activated. Thus, Monet programs are constructed entirely out of components which communicate via connections. Preferably, components do not communicate in any other manner, so that modular decomposition of program functionality is very explicit and very simple.

Although both Williams and the present invention are based on common dataflow notions and constructs, the patentable parts of each are the specific characteristics of each language and the development process defined for building software applications with it.

The following are some major differences between the present invention and Williams:

In claim 1 of Williams, a "program component," corresponding to the "process model" of the present invention,

has properties, some of which "surface" as "port connections," corresponding to the "slots" of the present invention.

By contrast, in the present invention a slot is not a property of a process model as in Williams, but rather a sub-component of a process model.

In Williams, connecting ports graphically is a mandatory step (step (e) in claim **1**). By contrast, in the present invention "flow rules" are optional components.

In Williams, the user needs to "surface" a property as a "port connection" to make it available for connection (see step (e)(i) of claim **1**). **In the present invention there is no "surfacing" step.**

The present invention includes several notions and constructs which do not exist in Williams:

Claim **1**: "Data models" (atomic and composite), which can be a sub-components of process models or other data models. Williams only describes properties of components.

Claim **1** has been amended by incorporating a portion of the language of claim **5**. The present invention system classifies input slots as mandatory or optional. This, when combined with the ability to model arbitrarily complex data structures, gives a significant expressive power, which is lacking in other dataflow languages.

Claim **7**: Mapping between database tables and data models.

Claim **10**: Various composition methods of composite data models (concatenation, collection, selection, recurrence).

As stated in the decision in In Re Marshall, 198 USPQ 344 (1978), "To constitute an anticipation, all material elements recited in a claim must be found in one unit of prior art…". Since the Williams reference neither 1) identically describes the invention, nor 2) enables one skilled in the art to practice it, Applicant deems the 102(b) rejection improper, and respectfully requests that it be withdrawn.

The dependencies of claims **2-12** and **14-16**, ultimately relating to claim 1, result in no amendments being needed.

The Examiner has rejected claim **13** under 35 U.S.C. 103(a) as being unpatentable over Williams, referring to column 2, lines 20-39, and column 5, line 31 to column 13, line 42. Again, since claim **13** of the present invention is indirectly dependent on claim **1** no further amendment is needed for claim **13**.

The Examiner has rejected claims **17-20, 24-25** and **27-28** under 35 U.S.C. 103(a) as being unpatentable over Williams, in view of Parr et al, (US20020095653).

Parr discloses a visual architecture software language to define discrete software modules and data flow and control flow between them. It includes a code generation step (called "compilation") to convert the visual representation into executable code in the form of executable instructions and data tables.

Code generation is a known technique to convert the definition of a software component (e.g. a software application) from one representation to another, typically from a high-level, user-friendly notation to a lower-level notation. Although both Parr and the present invention use code generation, the patentable parts of each are the specific characteristics of each language and the code generation algorithm used to generate code from it.

The code generator of the present invention is inherently related to the visual modeling language of the present invention, and it has to create code that is functionally 100% equivalent to the operation performed by the runtime engine described in claim 17 of the present invention. None of the features of this complex code generator can be inferred, logically or practically, from the mere notion of "compilation" mentioned by Parr.

Note that Parr discusses code generation in very general terms without explaining what "compilation" means (Parr's claim 16 is just half a sentence that says nothing on the generated code: "...compiling the visual representation to generate executable code, said executable code comprising").

The Examiner has rejected claims **21 - 23** under 35 U.S.C. 103(a) as being unpatentable over Williams, in view of Goodwin et al (US 6,199,195).

Goodwin deals with "automatically generated object-oriented source code." This is not the case with the code generator of the present invention.

Furthermore, the essence of the present invention is the completeness and preciseness of the language, which enable the needed code generation.

Although both Goodwin and the present invention use code generation, the patentable parts of each are the specific characteristics of each language and the code generation algorithm used to generate code from it.

Therefore, the code generator of the present invention is inherently related to the language of the present invention.

The Examiner has rejected claims **26** and **29** under 35 U.S.C. 103(a) as being unpatentable over Williams, in view of Parr et al, in further view of Goodwin et al.

It is respectfully put forward by the Applicant that there is no reason to consider the combination of prior art references to Williams, Parr and Goodwin as rendering the present invention unpatentable, since they include no provision for totally eliminating the need for writing code in any programming language to implement software applications.

LEGAL ARGUMENTS

It is the Applicant's position that the combination of the Williams, Parr and Goodwin references to form the basis of the Sec. 103(a) rejection is improper, and Applicant respectfully requests that it be withdrawn.

Therefore, claims 1 - 30 are deemed to be patentable.

With regard to the combination proposed by the Examiner, the legal standard for this combination has not been met by the Examiner, based on the In Re Lintner, In Re Regel, and In Re Clinton decisions.

The legal arguments against the Sec. 103 rejection are:

In Re Lintner (172 USPQ 560, 562, CCPA 1972):

"In determining the propriety of the Patent Office case for obviousness in the first instance, it is necessary to ascertain whether or not the reference teachings would appear to be sufficient for one of ordinary skill in the relevant art having the references before him to make the proposed substitution, combination or other modification."

In Re Regel (188 USPQ 136, CCPA 1975):

"...even if all the elements of a claim are disclosed in various prior art references, the claimed invention taken as a whole cannot be said to be obvious without some reason given in the prior art why one of ordinary skill would have been prompted to combine the teachings of the references to arrive at the claimed invention."

In Re Clinton (188 USPQ 365 CCPA 1976):

"do the references themselves... suggest doing what appellants have done".

In addition to the legal arguments referenced above, several other important legal arguments are presented based on the following decisions:

As stated in Application of Wesslau, 353 F.2d 238, 241 (CCPA 1965):

"It is impermissible within the framework of Sec. 103 to pick and choose from any one reference only so much of it as will support a given position,to the exclusion of other parts necessary to the full appreciation of what such reference fairly suggests to one of ordinary skill in the art."

As stated in Grain Processing Corp. v. American Maize-Products Corp., 840 F.2d 902. 908 (Fed. Cir. 1988):

"Care must be taken to avoid hindsight reconstruction by using the patent in suit as a guide through the maze of prior

art references, combining the right references in the right way so as to achieve the result of the claims in suit."

As stated in In Re Dance, 160 F.3d 1339, 1343 (Fed. Cir. 1998), before prior art references can be combined or modified, there must be some suggestion or motivation found in the art to make the combination or modification.

The Examiner is relying on speculation and hindsight reconstruction of the references in view of the invention, and the Examiner is using an arbitrary combination of references.

The only motivation for the combination suggested by the Examiner is provided by the Applicant's invention. The Applicant is the first to totally eliminate the need for writing code in any programming language to implement software applications.

It is respectfully put forward by the Applicant that there is no reason to consider the prior art references either individually or in combination, as rendering the invention obvious.

Based on the amendments to the claims and the above remarks, Applicant believes that the invention is novel and inventive and that all the pending claims in the application are deemed to be allowable. It is believed that the amendments do not raise any new issues which would require a further search by the Examiner. Further reconsideration and allowance of the application is respectfully requested at an early date.

In view of the foregoing remarks, all of the claims in the application are deemed to be allowable. Further reconsideration and allowance of the application is respectfully requested at an early date.

Respectfully submitted,

Edward Langer, Pat. Atty.
Attorney for Applicant
Reg. No. 30, 564

Shiboleth, Yisraeli, Roberts and Zisman LLP
1 Penn Plaza, Suite 2527, New York, NY 10119
Tel.: 212-244-4111 Fax.: 212-563-7108
423123